A New Approach to Commutative Watermarking-Encryption

Roland Schmitz[1], Shujun Li[2], Christos Grecos[3] and Xinpeng Zhang[4]

[1] Stuttgart Media University, Germany
[2] University of Surrey, UK
[3] University of the West of Scotland, UK
[4] Shanghai University, China

Abstract. We propose a new approach to commutative watermarking-encryption (CWE). A permutation cipher is used to encrypt the multimedia data, which leaves the global statistics of the multimedia data intact. Therefore, any *non-localized* watermarking scheme that depends only on global statistics of the multimedia data can be combined with the permutation cipher to form a CWE scheme. We demonstrate this approach by giving a concrete implementation, which manipulates the global histogram to achieve watermark embedding/detection.

1 Introduction

Encryption and watermarking are both important tools in protecting digital contents, e.g. in digital rights management (DRM) systems. While encryption is used to protect the contents from unauthorized access, watermarking can be deployed for various purposes, ranging from ensuring authenticity of content to embedding metadata, e.g. copyright or authorship information, into the contents.

The concept of commutative watermarking-encryption (CWE) was discussed in [1] with special emphasis on watermarking in encrypted domain. Four properties about watermarking in encrypted domain are formulated in [1, Sec. 2.2]:

- **Property 1.** The marking function $\mathcal{M}$ can be performed on an encrypted image.
- **Property 2.** The verification function $\mathcal{V}$ is able to reconstruct a mark in the encrypted domain when it has been embedded in the encrypted domain.
- **Property 3.** The verification function $\mathcal{V}$ is able to reconstruct a mark in the encrypted domain when it has been embedded in the clear domain.
- **Property 4.** The decryption function does not affect the integrity of the watermark.

As is pointed out in [1], Properties 2 and 3 are equivalent, if the encryption function $\mathcal{E}$ and the marking function $\mathcal{M}$ commute, that is,

$$\mathcal{M}(\mathcal{E}_K(I), m) = \mathcal{E}_K(\mathcal{M}(I, m)) \tag{1}$$

where $\mathcal{E}$ is the encryption function, K is the encryption key, I is the plaintext media data and m is the mark to be embedded.

Previous approaches to CWE are essentially based on one of the following two techniques: *Homomorphic Encryption*, where the encryption function is commutative to some basic arithmetic operations like addition or multiplication that can support a further watermarking step, or *Partial Encryption*, where only a part of the multimedia data is encrypted and the remaining data are watermarked. In the present contribution we propose a novel approach, namely to use a cipher in the sense that it encrypts the multimedia data fully but leaves some global properties untouched which are then used to embed the watermark. As a proof of concept of this new approach, we propose a CWE scheme for digital images by combining a permutation based cipher and a "non-localized" watermarking scheme working with the global image histogram in the spatial domain.

The rest of the paper is organized as follows. Previous work on CWE, histogram-based watermarking and joint encryption-watermarking are reviewed in Sec. 2. In Sec. 3 we describe our proposed CWE framework in greater detail. In Secs. 4 and 5 we analyze the security and computational complexity of our proposed CWE scheme. In Sec. 6 we show some experimental results. We conclude the paper in Sec. 7, where we also give some directions for further research.

2 Related Work

2.1 Commutative Watermarking-Encryption

One approach to commutative watermarking is provided by deploying homomorphic encryption techniques so that some basic algebraic operations such as addition and multiplication on the plaintexts can be transferred onto the corresponding ciphertexts, i.e., they are transparent to encryption [1, Sec. 2.1]. Especially, if both the encryption and the watermarking process consist of the same homomorphic operation, one gets a commutative watermarking-encryption scheme. Examples of homomorphic operations are exponentiation modulo n, multiplication modulo n and addition modulo n (including the bitwise XOR operation). One major drawback of this approach is the influence of encryption on robustness of the watermarking algorithm: After strong encryption there is no visual information available for the watermark embedder to adapt itself to in order to increase robustness while at the same time minimizing visual quality degradation [2, Sec. 9.4]. Another drawback is that the modular addition operation may cause overflow/underflow pixels that have to be handled separately, thus making the system "quasi-commutative" [3]. The XOR operation does not suffer from the overflow/underflow problem, though.

In partial encryption schemes, the plaintext multimedia data is partitioned into two disjoint parts, where one part is encrypted and the other part is watermarked. Since the encryption part is independent of the watermarking part, they are naturally commutative. To take a typical example, in [4], the multimedia data is partitioned into two parts after a four-level discrete wavelet transformation. The lowest-level coefficients are fully encrypted, while in the medium- and high-level coefficients only the signs are encrypted. In this case, the unencrypted absolute values of medium-level coefficients can be watermarked either before

or after encryption (if after, without access to the encryption key). However, there is a certain danger that an attacker might tamper with the encrypted, un-watermarked part. Depending on the encryption algorithm used, this might go unnoticed by the recipient.

Because there is some information leakage through the unencrypted parts, in order to get a high level of perceptual security, the data parts which are significant for perception are encrypted, while only the perceptually unimportant parts are watermarked, leaving the door open for an attacker trying to remove the watermark. In order to overcome these difficulties, in another recent proposal [5], a key-dependent transform domain, the Fibonacci-Haar transform, is used for both watermarking and encryption to increase protection for the unencrypted, watermarked part. After a first-order Fibonacci-Haar transform, the LL subband of each color component is fully encrypted. The remaining detail subbands are then watermarked. The main drawback of this approach is that neither decryption nor watermark detection is possible without knowledge of the key for the Fibonacci-Haar transform, which means that Property 3 cannot be fulfilled. Thus, by adding another layer of encryption, the original commutativity property of watermarking and encryption is lost. We call such schemes *joint watermarking-encryption* (JWE) to differentiate them from CWE schemes.

Thus, for both approaches to CWE there is a lack of robustness against malicious attacks, if strong encryption is used. This seems to be a general problem with CWE schemes (see also Sec. 4.1).

2.2 Asymmetric Joint Watermarking-Encryption

A very interesting approach is put forward in [6], where a permutation-based cipher is combined with an additive watermarking scheme acting on the 25×25 upper left corner of the DCT coefficients of an image. This scheme is truly asymmetric in the sense that different keys are used for embedding and detection of the watermark. Detection of the watermark in the encrypted domain is possible because the public key D used for detection contains some side information on the watermarked feature ψ and the encrypted watermarked feature ξ. This scheme is not a CWE scheme, however, because the watermark detection requires information on the encryption process.

2.3 Histogram Based Information Hiding

In [7] it is shown how a reversible information hiding scheme can be built by hiding data within the histogram of an image. The basic idea is to shift the grey levels of all pixels having a grey level between $g_{\min}$ and $g_{\max}$ towards $g_{\min}$, where $g_{\min}$ and $g_{\max}$ denote the grey level with the lowest and the highest heights in the histogram, respectively. Such a shift will make the histogram bin at the position $g_{\max} + 1$ or $g_{\max} - 1$ empty, thus "making space" for the data to be hidden.

2.4 Histogram Based Watermarking Schemes

The most widely studied approach to histogram based watermarking is so-called exact histogram specification [8–11], where the histogram of the original image or a (randomly and secretly selected) sub-region of it is modified toward a target histogram, which is then used as the signature for watermark detection. The histogram is not limited to be the one built from pixel values, but can also be a 2-D or 3-D histogram built from other features of the image [9–11]. To minimize visual quality distortion caused by the histogram manipulation, an optimization model can be used to find a globally optimum solution as demonstrated in [10].

However, most histogram based information hiding schemes cannot be used for secret watermarking because they do not involve a secret embedding/detection key. In what follows, we describe one approach that does use a secret watermarking key and whose basic principle is used in the example implementation of our proposed CWE framework.

The scheme proposed by Chrysochos et al. [12] is based on the idea of (selectively) swapping two selected neighboring histogram bins a and b so that a message bit is encoded by the heights of the two bins (denoted by hist(a) and hist(b)): a 1-bit is encoded by hist(a) $>$ hist(b) and a 0-bit by hist(a) $<$ hist(b). Here, swapping two histogram bins a and b means changing all pixel values a to b and vice versa. In order to embed an N-bit watermark into a 8-bit grey-level image, a watermarking key composed of a bin distance $1 \leq \text{step} \leq 9$ and a start bin index $0 \leq a_1 \leq 255 - \text{step}$ is needed. The i-th bin pair is selected by increasing a_1 by i but skipping those bin pairs breaking at the right boundary of the histogram. As the pixel values are changed by an amount of step when embedding the watermark, the step is upper bounded to nine in order to limit visual quality degradation. The embedding capacity of the scheme depends on the number of candidate histogram bin pairs whose heights are not equal (which is dependent on the image and the step), but it is bounded by 128 bits for 8-bit grey-level images and 384 bits for RGB images. Besides these low capacity bounds, the main problem of this scheme is the very small key space, which contains only $\sum_{\text{step}=1}^{9}(256 - \text{step}) = 2259$ different watermarking keys.

3 The Proposed CWE Framework

In order to design a CWE scheme, the encryption/decryption function must keep some features of the original image free from distortion so that they can be used for watermark embedding either before or after encryption. For instance, homomorphic encryption preserves the locations of all pixels so that the watermark embedding process can still happen on the intended pixels as long as the embedding function is commutative to the encryption function. Partial encryption preserves both locations and pixel values of part of an image so that watermark embedding can happen without any constraints. Neither of the existing approaches to CWE tries to preserve pixel values and distorts locations of all pixels. This led us to propose a third approach for designing a CWE scheme:

using a permutation cipher to encrypt the image to preserve all pixel values intact for watermarking embedding. What a permutation cipher does is to simply shuffle the locations of all pixels under the control of a secret key. Although no change is made to any pixel value, the ciphertext image normally looks random enough to achieve the goal of concealing almost all visual information carried by the original image. Since no pixel value is changed by a permutation cipher, the global histogram of the image remains intact. If a watermarking scheme only uses the global histogram of the image for embedding and detection, which we call *non-localized watermarking*, the permutation (as an encryption function) and the watermarking processes will become commutative, satisfying all the four properties listed in Sec. 1. Examples include the image watermarking schemes proposed in [10, 12] and the video watermarking scheme proposed in [13]. The last scheme makes use of both the spatial histograms of single frames and the temporal histogram for a given video sequence.

An obvious advantage of this new approach is that the robustness of the watermarking algorithm remains intact because all information (global statistics) required by the algorithm is not changed by encryption. In this aspect, the new approach outperforms the homomorphic cryptography based CWE approach. Compared with the partial encryption based approach, our proposed scheme can provide a higher level of security since total encryption is applied here.

3.1 Watermarking Part

The watermarking part is designed following the basic principle of the histogram based watermarking scheme proposed in [12]. However, since this scheme suffers from two severe limitations, namely a very small key space and a small capacity, we have devised a modified watermarking scheme to overcome both problems.

Basic Scheme For embedding the watermark, we select each bin pair randomly from all remaining candidates rather than in a sequential order as in the original scheme [12], which leads to a significantly bigger key space. The process is driven by a stream cipher that serves as a secret pseudo-random number generator. The watermark is encrypted so that the order of selected bin pairs matters in the extraction of the watermark. Given an N-bit watermark, the bin pairs selection, watermark embedding and detection processes can be described as follows.

Bin pairs selection: For the i-th bin pair, run the stream cipher to create a random integer $0 \leq x \leq 255 - 2i$. Then pick the x-th unused bin as the first bin a_i. Then, run the stream cipher to create a new integer $\max(-9, -a_i) \leq \text{step} \leq \min(255 - a_i, 9)$. Pick the $(a_i + \text{step})$-th bin as the second bin b_i. If b_i has been used or if the two bins have the same height, re-generate a new integer x and a new step until two valid bins are selected to form a new bin pair.

Watermark embedding: First encrypt the watermark $W = \{w_i\}_{i=1}^{N}$ by the stream cipher to get $W^* = \{w_i^*\}_{i=1}^{N}$. The heights of the two selected bin pairs a_i and b_i should encode w_i^* as follows: if $w_i^* = 1$, $\text{hist}(a_i) < \text{hist}(b_i)$ should hold, and if $w_i^* = 0$, $\text{hist}(a_i) > \text{hist}(b_i)$ should hold, where $\text{hist}(x)$ denotes the height of the bin x. If this is not the case, the two bins a_i and b_i are swapped.

Watermark extraction: First, reconstruct the same sequence of bin pairs $\{a_i, b_i\}_{i=1}^{N}$ at the detector side. Then, extract the encrypted watermark as follows: $W^* = \{w_i^*\}_{i=1}^{N}$, where $w_i^* = 0$ if $\text{hist}(a_i) > \text{hist}(b_i)$ and $w_i^* = 1$ if $\text{hist}(a_i) < \text{hist}(b_i)$. Finally, decrypt W^* to recover the plaintext watermark W.

Figure 1 shows the results of embedding a 64-bit watermark "12345678" into the blue channel of the test image "baboon" by using the modified watermarking scheme and by the original watermarking scheme.

(a) (b) (c)

Fig. 1: Embedding a 64-bit watermark "12345678" into an image's blue channel: (a) original image; (b) image watermarked by the modified scheme (PSNR = 42.57); (c) image watermarked by the original scheme (PSNR = 42.36).

We ran both watermarking schemes on the Kodak true-color image database and measured the quality of the watermarked images by using ten objective visual quality assessment (VQA) metrics included in the MeTriX MuX VQA Package [14]. The results show that our changes to the original scheme does not compromise the visual quality of the watermarked image. To be more exact, the mean of the visual quality measured by all the ten VQA metrics remains similar for both schemes but our scheme seems to have a smaller variance in the measured visual quality, which can be partly explained by the stronger random effect of the bin selection process. Figure 2 shows the PSNR and SSIM (two VQA metrics) values of 24 images watermarked by the two schemes.

Enhancing the Capacity The capacity of the basic scheme described above is limited to the number of candidate bin pairs, which is upper bounded by 128 bits. It can be greatly enhanced by dividing the cover work into sub-images and applying the basic scheme to those sub-images independently. In order to keep the visual distortions at a level comparable to that of the basic scheme, the sub-images should have roughly similar histogram shapes as the underlying image. This can be achieved either by randomly assigning image pixels to sub-images or by doing this using a predefined fixed pattern, where each pixel in an $n \times m$ block is assigned to one of $n \times m$ sub-images. Both approaches yield histograms similar to the original one. For simplicity reasons, we chose the latter approach in our prototype implementation. More specifically, for a pixel $p(i, j)$ in the original

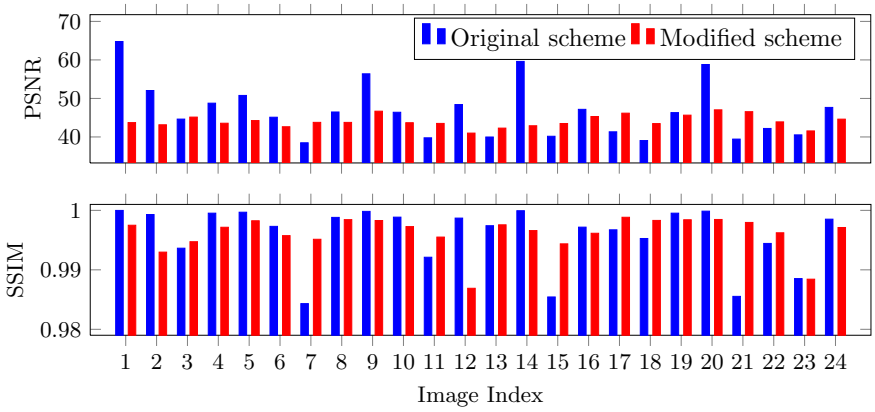
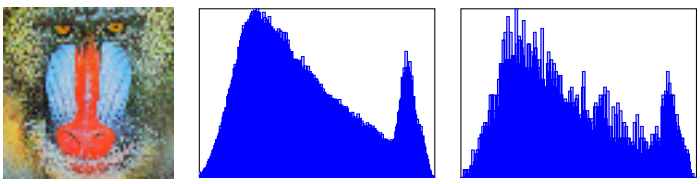


Fig. 2: Visual quality comparison of the modified watermarking scheme and the original one, measured by PSNR and SSIM.

image I, we compute $k = i \mod n$ and $\ell = j \mod m$ and assign $p(i, j)$ to sub-image $S(k, \ell)$. Figure 3 shows one resulting sub-image and the corresponding histograms for the blue channel of the baboon image in the case $n = m = 8$.

The maximum capacity achievable by this approach depends on the size of the sub-images. For our prototype, we chose the sub-images to be $s \times s$ images, where s is a common divisor of width W and height H of the underlying image. This choice was motivated by our use of Arnold's cat Map for encrypting the image (see Sec. 3.2), but in principle non-square sub-images are also possible. Moreover, the sub-image size does not need to be a divisor of W and H. In the most general case, the maximum capacity per colour channel is $C_{\max}(P) = 128 \cdot \lfloor WH/P \rfloor$ bits, where P is the number of pixels in one sub-image.

In our prototype implementation, we set $s \geq 50$ to ensure a meaningful histogram of each sub-image. Thus, here the overall maximum capacity is $C_{\max} = 128 \cdot \lfloor WH/s^2 \rfloor$ bits per colour channel, where s is the smallest common divisor of W and H that is ≥ 50. See Table 2 for some experimental results.



(a) (b) (c)

Fig. 3: Splitting the cover work into 64 parts: (a) a sub-image; (b) histogram of the original image; (c) histogram of the sub-image (a).

3.2 Encryption Part

Permutation ciphers have been very popular in securing analog Pay-TV services [15] and digital multimedia data in general [16] because they can be easily implemented and perceptual information about the ciphertext can be effectively concealed. A permutation cipher acting on an $W \times H$ image can be modeled by a $W \times H$ permutation matrix $\boldsymbol{M} = \{m(x,y) = (x',y')\}_{\substack{0 \le x,x' \le W-1 \\ 0 \le y,y' \le H-1}}$, where (i',j') denotes the new location of the pixel (i,j) after permutation [17, Sec. 2].

Note that the same permutation matrix can be used for encryption and decryption because the permutation is always a bijection. In principle, one can use the permutation matrix as the secret key, however, which occupies too much space so that the key management becomes difficult. A common practice is to use an algorithm to generate a permutation matrix under the control of a few parameters, which are used as the secret key. One of the simplest algorithms is as follows: generate a sequence of WH random numbers, then sort them, and finally take the 1-D indices which can be converted into 2-D coordinates to form the permutation matrix. Here, the random sequence can be generated by a stream cipher so that the permutation matrix is secret. The main drawback of this simple algorithm is about its complexity: the average complexity of a fast sorting algorithm is $O(WH \log_2(WH))$ and the worst-case complexity is $O((WH)^2)$ [18]. While the complexity is actually not very high, when WH is large, the factor $\log_2(WH)$ can still be significant. For instance, for full HD videos $\log_2(WH) = \log_2(1920 \times 1080) \approx 21$.

Many researchers have suggested iterating a parameterized 2-D discrete map to generate the permutation matrix. The average and worst-case complexities of such an approach is both $O(nWH)$, where n is the number of iterations. If the image size is known in advance, the permutation matrix in each iteration can be pre-computed, thus leading to a reduced computational complexity of $O(WH)$.

For our prototype implementation of the proposed CWE framework, we choose Arnold's cat map [19], which was used by several researchers for encrypting square images [20, 21]. Non-square images have to be either padded to be a square image or decomposed into a union of smaller square sub-images, like we did in the enhanced watermarking scheme described in Sec. 3.1. Arnold's cat map in its original form is defined on the unit square by

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} \mod 1, \tag{2}$$

where "mod 1" means taking the fractional part of the argument.

Given an $H \times H$ image, one discretized version [21] is defined as follows:

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} \mod H, \tag{3}$$

where a and b are parameters that can serve as the secret key if the function is used for encryption purposes. Figure 4 shows the results of applying Arnold's

(a) (b) (c) (d)

Fig. 4: Encryption results of the permutation cipher based on Arnold's cat map: (a) and (c) plain images; (b) and (d) ciphered images.

cat map to the test images "baboon" and "parrots", respectively. The baboon image was encrypted in its original form, while the parrots image was subdivided into 2×3 square sub-images before encryption. After that, each sub-image was encrypted using a different key.

3.3 Optional Information Hiding Part

Histogram based data hiding schemes like the one in [7] are localized due to the need of bookkeeping the locations of some pixels, therefore, they cannot be used for watermarking in the context of our applications. However, they may still be used for the encryption part, e.g. to transport part of the key and other meta-information needed for the decryption process (cf. Sec. 4.2).

4 Security Analysis

As the watermarking and encryption schemes deployed are completely independent, they do not interfere with each other and their security can be assessed separately. Further, we can restrict our analysis to the basic scheme, as the sub-images are watermarked and encrypted independently from each other.

4.1 Watermarking Part

Unauthorized Embedding and Detection The watermarking scheme described in Sec. 3.1 is driven by a stream cipher selecting the candidate histogram bin pairs for embedding. The number of all possible selections of different bin pairs, denoted by $S(N)$, depends on N, the length of the embedded watermark W. The size of the key space is therefore $\min(2^{|K|}, S(N))$, where K is the key of the stream cipher and $|K|$ is the its bit size.

In this subsection we derive two lower bounds on $S(N)$, which correspond to two different ranges of the length of embedded watermark N. To simplify our discussion, we assume W is a sequence of bits $W = b_0 b_1 \dots b_{N-1}$, where $N \geq 8$.

1. *Case 1:* $8 \leq N < 20$. Since $S(N_1) < S(N_2)$ if $N_1 < N_2$, we calculate the lower bound for $N = 8$. We limit ourselves to those histogram bins with 18

neighbors to make our calculation easier. There are 256-18 such bins. After having embedded i watermark bits, $2i$ bins have already been used. Likewise, at most $2i$ bins in the neighborhood of the first selected bin for embedding b_{i+1} are occupied by previously selected bins. Combining these two facts, we immediately have the following lower bound:

$$S(N) > \prod_{i=0}^{7}((256-18)-2i)\times(18-2i)) \approx 2^{89}. \tag{4}$$

2. *Case 2:* $20 \leq N \leq 128$. A lower bound can be obtained by considering only a subset of all the possible keys. A simple subset can be obtained as follows (without loss of generality, assuming 256 can be divided by N): partition all the 256 bins into N disjoint parts of equal size $256/N$, then randomly select a permutation of the N parts, and finally randomly pick two bins in each part whose distance is not greater than nine grey levels. Since in each part the number of bins may be smaller than nine, there are $\left(\frac{256}{N}\cdot\min\left(9,\frac{256}{N}-1\right)\right)$ possibilities for each part. Thus, we get the following lower bound:

$$S(N) > N!\cdot\left(\frac{256}{N}\cdot\min\left(9,\frac{256}{N}-1\right)\right)^{N}. \tag{5}$$

For $N = 32$, the lower bound is already well beyond 200 bits key length.

Unauthorized Removal Unfortunately, the watermark cannot withstand a malicious attacker manipulating the image histogram. The watermark may be removed by either randomly swapping neighbouring histogram bins or shifting the whole histogram by a small amount. This problem seems unavoidable since such attacks can simply resemble the original embedding algorithm.

Robustness Histogram based watermarks are known to be resistant against geometric attacks since the histogram is largely invariant to geometric transformations. More precisely, according to [22], the histogram is preserved by image transformations $\Psi_t : D \to \mathbb{R}^2$, where $D \subset \mathbb{R}^2$ is the domain of the image and $t \in \mathbb{R}$ is the parameter of the transformation, with the property $\operatorname{div}\left(\frac{d}{dt}\Psi_t\right) = 0$.

When working in the encrypted domain, most signal processing operations can be ruled out in the robustness discussion, as the resulting image cannot be decrypted anymore. However, images encrypted by a permutation cipher can be lossily compressed [23]. Robustness against this kind of compression will be subject of further research.

4.2 Encryption Part

It is well known that pure permutation-based ciphers are vulnerable to known- and chosen-plaintext attacks. A quantitative study was reported in [17], where it is shown that for an $H \times H$ square image with L grey levels $O(\log_L H^2)$

known plaintexts are sufficient to recover half of the plaintext pixels. The computational complexity of these attacks is $O(p \cdot H^4)$, where p is the number of known ciphertexts used, making these attacks practical. Therefore, we propose to use image-varying keys, e.g. image-dependent keys derived from the (normal or visual) hash of the image. The key is divided into one long-term secret master key and one short-term public image-dependent session key. The latter can be embedded into the encrypted image by using a reversible information hiding scheme such as those described in Sec. 2.3. It is combined with the secret master key to form the key for decrypting the image.

Arnold's cat map used in our prototype implementation suffers from a small key space of H^2, if the same parameters are used for all iterations. In addition, it is well known that any discrete area-preserving map is periodic (upper bounded by the total number of finite states) and the discrete cat map applied to binary images has a period upper bounded by $3H$ [24]. For some "bad" parameters, the period can be very short. For example, selecting $a = 40, b = 8$ yields the original image again after five iterations if $H = 124$ [25].

The security problems with Arnold's cat map can be mitigated by using different keys for different iterations, which can increase the key space to H^{2n} and also reduce the influence of bad parameters on the final result. We generated 1000 random keys with parameters $H = 124$ and $n = 20$, and none of the generated permutation matrix degenerates to the identity matrix. This led us to believe that the combination of n different keys is strong enough for our purpose.

If the sorting based approach mentioned in Sec 3.2 is used to generate a random permutation matrix, the security problems with Arnold's cat map will disappear. In this case, the key space becomes $(H^2)!$ and the short period does not exist anymore since we stop depending on iterating a 2-D map repeatedly.

5 Complexity Analysis

Without loss of generality, we assume that the plaintext image is an $H \times H$ image with L grey levels, that the watermark is an N-bit pattern, and that N is much smaller than H^2, so that the derived complexity can be more compact. As in Sec. 4, we restrict the analysis to the basic scheme with no sub-images. In addition, we only consider the average complexity because the worst-case complexity can be quite different and less meaningful.

5.1 Watermarking Complexity

Generating the histogram corresponds to H^2 operations. To select N bin pairs from the histogram, $\approx 2N$ operations are needed. To embed all the N bits, averagely $N/2$ bin pairs need swapping, whose complexity is NH^2/L. To detect the watermark, only N comparisons of bin heights are needed. To sum up, the overall computational complexity of the watermark embedding process is $O(NH^2/L)$ and that of the watermark detection process is $O(H^2)$.

5.2 Encryption/Decryption Complexity

Since any permutation cipher can be represented by an $H \times H$ permutation matrix, encrypting an image requires merely H^2 look-up table operations and H^2 assignments. Iterating the discrete 2-D cat map n times requires nH^2 look-up table operations and pixel value assignments. Generating the permutation matrix requires $3H^2$ multiplications and $3H^2$ addition/assignments. We can ignore the $3H^2$ additions/assignments, because multiplications are computationally much heavier. Thus, the overall complexity becomes $O((n+5)H^2) = O(nH^2)$. When the sorting based approach is used to generate the permutation matrix, the overall complexity is $O(\log_2(H)H^2)$. Since the decryption can be done by using the same matrix, the computational complexity remains the same.

5.3 Comparison with Existing Schemes

Table 1 shows the complexities of our proposed CWE scheme and some existing ones following the other two approaches to CWE.

6 Experimental Results

Table 2 shows the results of applying the proposed CWE scheme including subdivision into square sub-images to four test images. By design, the watermarked-encrypted image and the encrypted-watermarked image are the same. In all images, a random bit sequence of maximum length was embedded into the blue channel. The PSNR and SSIM values were calculated by comparing the blue channels of the cover work and the marked image.

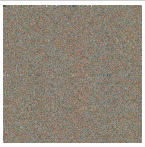
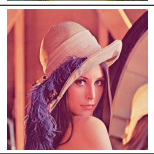
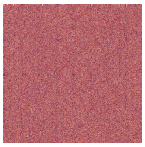
The watermark could be successfully extracted either from the encrypted marked image $\mathcal{E}_K(\mathcal{M}(I,m))$ (Property 3) or from the marked encrypted image $\mathcal{M}(\mathcal{E}_K(I),m)$ (Property 2). In all cases, decrypting either $\mathcal{M}(\mathcal{E}_K(I),m)$ or $\mathcal{E}_K(\mathcal{M}(I,m))$ leads to the marked plaintext image $\mathcal{M}(I,m)$, from which the watermark could still be successfully extracted (Property 4).

Table 1: Complexities of our CWE proposed scheme and some existing ones.

Scheme	Watermarking embedding complexity	Watermarking detection complexity	Encryption complexity
Proposed CWE scheme*	$O((N/L+1)H^2)$	$O(H^2)$	$O(nH^2)$
Homomorphic CWE schemes in [2, Sec. 9.3]	$O(H^2)$	$O(H^2)$	$O(H^2)$
Partial encryption based CWE scheme in [4]**	$O(mH^2)$	$O(mH^2)$	$O(mH^2)$

*: n is the number of iterations of the cat map.
**: m denotes the length of the low-pass and high-pass wavelet filters.

Table 2: Some experimental results.

Plain Image I	$C_{\max}$ $n \times m$*	Marked Image $\mathcal{M}(I,m)$	PSNR SSIM	$\mathcal{E}_K(\mathcal{M}(I,m)) = \mathcal{M}(\mathcal{E}_K(I),m)$
	8192 8×8		36.69 0.966	
	8192 8×8		36.56 0.913	
	12288 8×12		36.45 0.876	

*: $n \times m$ denotes the size of the sub-image array

7 Conclusion and Further Work

We have presented a novel approach to building commutative watermarking-encryption schemes based on permutation-only ciphers and non-localized watermarking schemes. A concrete CWE scheme was designed by combining a histogram based watermarking scheme with a permutation cipher based on a discrete 2-D chaotic map. It satisfies all the four properties formulated in [1]. Due to its simplicity, the proposed scheme is well suited for applications with high performance requirements, such as video content protection or authenticity of large amounts of encrypted transcoded data in heterogenous networks.

In our future work, we will study a possible generalization of the proposed CWE scheme to compressed domain, where the key questions include how to apply permutations without compromising compression efficiency and how to make the watermarking scheme more robust to lossy compression. We will also investigate if reversible CWE schemes can be designed within the proposed framework.

References

1. Herrera-Joancomartí, J., Katzenbeisser, S., Megías, D., Minguillón, J., Pommer, A., Steinebach, M., Uhl, A.: ECRYPT European Network of Excellence in Cryptology, first summary report on hybrid systems, D.WVL.5 (2005)
2. Lian, S.: Multimedia Content Encryption. CRC Press (2009)
3. Lian, S.: Quasi-commutative watermarking and encryption for secure media content distribution. Multimedia Tools and Applications **43**(1) (2009) 91–107
4. Lian, S., Liu, Z., Zhen, R., Wang, H.: Commutative watermarking and encryption for media data. Optical Engineering **45**(8) (2006)

5. Battisti, F., Cancellaro, M., Boato, G., Carli, M., Neri, A.: Joint watermarking and encryption of color images in the Fibonacci-Haar domain. EURASIP J. Advances in Signal Processing **2009** (2009) Article ID 938515
6. Boato, G., Conotter, V., DeNatale, F.G.B., Fontanari, C.: A joint asymmetric watermarking and image encryption scheme. In: Security, Forensics, Steganography, and Watermarking of Multimedia Contents X. Volume 6819 of Proc. SPIE. (2008) 68191A
7. Ni, Z., Shi, Y.Q., Ansari, N., Su, W.: Reversible data hiding. IEEE Trans. Circuits and Systems for Video Technology **16**(3) (2006) 354–361
8. Coltuc, D., Bolon, P.: Robust watermarking by histogram specification. In: Proc. 1999 Int. Conf. Image Processing (ICIP'99). Volume 2. (1999) 236–239
9. Chareyron, G., Coltuc, D., Trémeau, A.: Watermarking and authentication of color images based on segmentation of the xyY color space. J. Imaging Science and Technology **50**(5) (2006) 411–423
10. Roy, S., Chang, E.C.: Watermarking color histograms. In: Proc. 2004 Int. Conf. Image Processing (ICIP 2004). (2004) 2191–2194
11. Lin, C.H., Chan, D.Y., Su, H., Hsieh, W.S.: Histogram-oriented watermarking algorithm: colour image watermarking scheme robust against geometric attacks and signal processing. IEE Proc. Vision, Image and Signal Processing **153**(4) (2006) 483–492
12. Chrysochos, E., Fotopoulos, V., Skodras, A.N., Xenos, M.: Reversible image watermarking based on histogram modification. In: Proc. 11th Panhellenic Conf. Informatics (PCI 2007). (2007) 93–104
13. Chen, C., Ni, J., Huang, J.: Temporal statistic based video watermarking scheme robust against geometric attacks and frame dropping. In: Proc. 8th Int. Workshop Digital Watermarking (IWDW 2009). Volume 5703 of Lecture Notes in Computer Science. (2009) 81–95
14. Gaubatz, M.: MeTriX MuX visual quality assessment package. `http://foulard.ece.cornell.edu/gaubatz/metrix_mux`
15. Slater, J.: Scrambler: TV signal encryption. Electronics Today Int. (ETI) **19** (1990) 16–20
16. Zeng, W., Lei, S.: Efficient frequency domain selective scrambling of digital video. IEEE Trans. Multimedia **5**(1) (2003) 118–129
17. Li, S., Li, C., Chen, G., Bourbakis, N.G., Lo, K.T.: A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. Signal Processing: Image Communication **23**(3) (2008) 212–223
18. Knuth, D.: The Art of Computer Programming, Volume 3: Sorting and Searching. 2nd edn. Addison-Wesley (1998)
19. Arnold, V.I., Avez, A.: Ergodic Problems of Classical Mechanics. Benjamin (1968)
20. Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. Int. J. Bifurcation and Chaos **8** (1998) 1259–1284
21. Chen, G., Mao, Y., Chui, C.K.: A symmetric image encryption scheme based on 3D chaotic cat maps. Chaos, Solitons and Fractals **21**(3) (2004) 749–761
22. Hadjidemetriou, E., Grossberg, M.D., Nayar, S.K.: Histogram preserving image transformations. Int. J. Computer Vision **45**(1) (2001) 5–23
23. Zhang, X.: Lossy compression and iterative reconstruction for encrypted image. IEEE Trans. Information Forensics and Security **6**(1) (2011) 53–58
24. Dyson, F.J., Falk, H.: Period of a discrete cat mapping. The American Mathematical Monthly **99**(7) (1992) 603–614
25. Wong, K.W.: Image encryption using chaotic maps. In: Intelligent Computing Based on Chaos, Springer (2009) 333–354